BCAUS PROJECT DESCRIPTION AND CONSIDERATION OF SEPARATION OF DATA AND CONTROL

by
Joy L. Bush and Steven J. Weaver
Computer Sciences Corporation

ABSTRACT

Our experience with the development of a diagnostic expert system has caused us to examine the commonly stated truths that data may be segregated from program control in "generic" expert system shells and that such tools support straightforward knowledge representation. We believe that the ideal of separation of data from program control in expert systems is difficult to realize for a variety of reasons. One approach to achieving this goal is to integrate hybrid collections of specialized shells and tools instead of producing custom systems built with a single "all purpose" expert system tool.

In this report, we try to examine aspects of these issues in the context of a specific diagnostic expert system application, the Backup Control Mode Analysis and Utility System (BCAUS), being developed for the Gamma Ray Observatory (GRO) spacecraft. We present a description of the project, our experience to date, and plans for the on-going system development. A more detailed description of the BCAUS prototype development appears in [1].

1 BCAUS Description

BCAUS is an expert system designed to assist flight operations personnel in diagnosing the cause of a GRO spacecraft autonomous mode transition. The GRO spacecraft has been designed with onboard capability to autonomously *safe* itself, transitioning from a primary operating mode to a backup control (safing) mode, in the event of certain error conditions in the Attitude Control and Determination (ACAD) subsystem. (The logic for the autonomous transitions is described in several papers [2,3].) Flight operations personnel need to understand what error condition triggered the onboard computer (OBC) to order the mode transition and why that error condition occurred so that proper corrective action may be taken. While the actual number of mode transition triggers is small, there are potentially hundreds of underlying causes that could affect these triggers. Thus, the task of diagnosing ACAD failures is nontrivial and involves substantial expertise.

The BCAUS system is intended for use in the GRO Mission Operations Room, where it will be resident on either a 386-PC microcomputer or a Silicon Graphics Iris 4D/20 computer. The initial prototype was developed on a 386-PC-equivalent microcomputer. Input to the system will consist primarily of telemetry from the spacecraft via processing operations within the Multisatellite Operations Control Center, and operator initialization and input. Output from the system will be to the operator only; no output to the spacecraft is involved.

Use of the operational version of BCAUS would typically be initiated when the flight operations team (FOT) realizes, upon achieving contact with GRO during a pass, that the spacecraft is in a backup control mode having made a transition autonomously while out of ground contact. Normally, operations personnel attempt to command the High Gain Antenna and acquire high-speed telemetry and a tape recorder dump. If successful, real-time telemetry is available immediately; the recorded telemetry is available after some processing delay. Once processed, the tape-recorded telemetry is available through the System Test and Operations Language interface, and BCAUS will access these data. It is expected that BCAUS will depend primarily on tape recorder dumps to obtain telemetry data for the diagnostic process rather than using real-time telemetry. This is because GRO has relatively brief periods of ground contact, so it is likely that a mode transition will occur when the spacecraft is not in contact. Also, very little information about the

mode transition is preserved in telemetry after the event, so it is necessary to access the recorded telemetry during the time period leading up to and during the mode transition.

The approach taken with BCAUS has been to develop a prototype knowledge base which encodes the expertise spacecraft subsystem engineers use to identify anomalies that trigger autonomous mode transitions. This approach is in contrast to an effort to model the operation and interaction of the spacecraft systems themselves. The focus of BCAUS is on the autonomous mode transitions that result from ACAD safety checks, which is a narrower problem domain than overall spacecraft health and safety. However, it has been necessary to consider the potential effects of failures in many subsystems on spacecraft attitude.

2 Knowledge Acquisition

The knowledge acquisition activities undertaken for the development of the BCAUS prototype involved the following: extensive review of GRO documentation; interviews with Goddard Space Flight Center (GSFC) personnel experienced in spacecraft design, oversight, and operations; interviews with GSFC contractor personnel experienced in spacecraft flight operations and engineering; and interviews with TRW personnel with GRO subsystem engineering, design, and software experience.

The GRO documentation was useful both as an introduction to GRO ACAD operation and as a resource that was revisited as task members accumulated an increased understanding of technical details. However, the most critical aspect in expert systems development is the capture of domain expertise from human experts. The BCAUS prototype is to provide expert assistance in the diagnosis of GRO spacecraft anomalies that lead to an autonomous mode transition, and so requires the input of persons knowledgeable in recognizing anomalous symptoms in GRO flight operations and tracing the causes of those symptoms. However, since GRO has not yet flown, no one yet has that particular expertise. Task members, therefore, sought the input of those persons most closely concerned with the development of the GRO spacecraft and those with flight operations experience on other similar spacecraft. TRW engineering personnel, based in California, possess the most intimate knowledge of the relevant GRO subsystems, and were thus a source of major importance. They were also the least accessible.

Lacking immediate access to the GRO spacecraft subsystem experts, BCAUS developers attempted to acquire sufficient technical knowledge to hypothesize potential failures and symptoms, rather than employing the usual approach of querying the experts for this information. The periods of interaction with the actual experts were then used to attempt to confirm or deny the developers' suppositions. While this has provided sufficient information for a reasonable prototype, more extensive access to experts will be necessary to produce an operational version.

3 Tools Chosen for Implementation

3.1 Prototype Tool Description

KES expert system development software, produced by Software Architecture and Engineering, Inc., was selected for the prototype development. The KES package includes three different expert system shells: a hypothesize-and-test inference engine (HT), a production rule system (PS), and a Bayesian probability inference engine (BAYES). We used the KES HT inference engine, which has the advantage of a built-in diagnostic reasoning technique, *minimal set covering*, which simulates a human-like, sequential, hypothesize-and-test process.

Because the KES HT knowledge base structure was developed for diagnostic applications, the entry of diagnostic knowledge was straightforward, using a cause-manifests-symptoms frame rather than an IF-symptoms-THEN-cause rule. Minimal set covering reasons very efficiently about multiple causes, eliminates impossible hypotheses, and focuses on the most likely hypothesis. The rapid reduction of the search space also generally reduces the number of questions generated by the system for the user to

answer. The KES HT implementation also offers a straightforward yet useful handling of confidence factors with reasonable defaults. It uses these confidence factors to either consider or reject hypotheses and to automatically provide a rank ordering of possible solutions from most to least likely.

In BCAUS, a set of symptoms is associated with each possible failure. In a diagnostic problem with a specific set of symptoms present, the KES HT inference mechanism finds all sets of failures that explain or *cover* the set of all symptoms. KES HT reports, as a result of the diagnostic operation, all sets of failures with minimum cardinality (that is, the hypotheses containing the minimum number of failures needed to explain the symptoms). A hypothesis that contains more than that minimum number is not included in the answer. The use of a minimal set cover is based on *parsimony*, the assumption that the simplest explanation is usually the correct one. (See [4] for a full discussion of the theory upon which KES HT reasoning is based.) The KES HT use of parsimony is in keeping with the general philosophy of failure analysis for the GRO project. Based on spacecraft reliability expectations, single-point failures are considered to have a low probability, and multiple-point failures are considered unlikely enough to be largely excluded from the failure analyses. This means that BCAUS will offer a multiple-failure diagnosis only if no single failure is known which can account for the observed symptoms.

In KES HT, diagnostic failure and symptom knowledge is represented in frames. Each frame contains knowledge about a class of failure. Essentially, a KES HT knowledge base description of a problem consists of the name of the problem and a description of the associated symptoms and their symbolic values. Also included is the likelihood of that particular symptom's appearing [i.e., the symbolic certainty factor (SCF)]. KES HT also provides for "setting factors" which are a variation of a symptom. A setting factor is a convenient way of indicating that the absence of a symptom does not eliminate the possibility of a problem, but that its presence makes the problem more likely (the degree of likelihood being indicated by an SCF). Task personnel made extensive use of setting factors to allow a telemetry mnemonic to have a normal (unstated) value in a specific failure case. By only having to explicitly specify non-normal values in the failure frames, failure frame size (and memory usage) was greatly reduced.

3.2 Prototype Implementation

The initial prototype knowledge base entry and debugging took approximately three calendar weeks for two persons. Because of the built-in diagnostic feature of KES HT, there were no rules to enter or debug. The prototype knowledge base contained approximately 100 frames describing possible causes for the eleven triggers.

When KES HT was selected as the software tool for prototype development, some limitations with the tool had been identified. The limitations of KES HT in the area of explanation and justification were more restrictive than anticipated. KES does provide the capability of attaching textual explanations to questions used to elicit user input and to values appearing on the question response menus. This capability was adequate for the purpose of explaining or enlarging on the questions and answers. There is no capability, however, to explain why a question is asked at any given time; the system cannot inform the user that the question's purpose is to confirm or deny a particular hypothesis, for instance. Likewise, while KES HT does provide a trace capability that permits the user to keep track of what hypotheses (potential failures) are under consideration, it does not provide the means to explain how the final conclusion was reached. It should also be noted that KES applications are limited to the 640 kilobyte address space under DOS.

In addition, the possibility exists that the minimal set covering process may eliminate sets of possible failures that would explain the symptoms in certain cases. When the reasoning process develops a number of possible explanations for certain symptoms, each explanation, by the parsimony assumption of minimal set covering, would have the same number of failures, say for example two. A possible explanation of the symptoms in this case that includes three failures would not be considered as a solution. The consideration of *irredundant* solutions by the KES HT algorithm would allow, as a solution, an explanation of the symptoms that included three or even more failures. (Irredundancy says that no set of

failures will be included in the solution if the set contains, as a subset, a set of failures that already exist as a solution. See [5] for further explanation of irredundancy.)

3.3 Operational System Implementation

ART-IM, a commercial, off-the-shelf expert system shell from Inference Corporation was selected for the development of the operational BCAUS. ART-IM is a relatively new product; a scaled-down version of the ART system, implemented in C, using the C Language Integrated Production System (CLIPS) as a basis. It is primarily a forward-chaining rule-based system, but provides object-like structures called schemas. A primary reason for this choice was our desire to maintain flexibility in the final choice of delivery platform. ART-IM can be used for development on the 386-PC, and delivered on either the 386-PC or the Silicon Graphics 4D/20.

There are a number of other reasons which contributed to the selection of ART-IM. Although the KES HT frame-based reasoning is very attractive, it carries a number of limitations, including the inability to reason about numeric values without having them transformed to symbolic values, and the lack of full explanation/justification, even with proposed enhancements. ART-IM offers schemas, which can be used in a manner similar to KES HT's frames. ART-IM does not possess the minimal-set-covering type of abductive reasoning that KES HT has, but it does provide rules with very powerful pattern matching capabilities which allows us to tailor operations acting on schemas to our application. The development environment is very good, providing a wide variety of tracing and browsing features.

In evaluating ART-IM as a candidate for BCAUS operational development, we created and ran a small diagnostic system, using schemas which represent disorders with symptoms, in conjunction with rules that acted upon the input symptoms and the schemas. It appeared that ART-IM is capable of implementing the kind of diagnostic system we want.

4 FUTURE DEVELOPMENT

4.1 Porting KES HT Knowledge Base to ART-IM

We are currently converting the KES HT failure frames to ART-IM schemas. We are also extending the previously described small diagnostic system to emulate minimal set covering in ART-IM rules. The emulation of minimal set covering in ART-IM rules has made extensive use of the schema operators provided in ART-IM. The inclusion of procedural control aspects within rules has implications for the development and maintenance of the expert system; this is discussed in Section 5.

4.2 Neural Network Front-end for Telemetry Trending Analysis

The need for having trending data available for the diagnostic task is clear. Many of the failure scenarios known to the knowledge base involve symptoms that require information on the behavior of a value, as opposed to a single reading. For instance, a "noisy" gyro cannot be recognized by one unusual reading. Rather it is the existence of several aberrant, or least inconsistent, readings that is the significant indication. We plan to add a facility to accumulate data over time with regard to specific telemetry mnemonics, and assess the trend of the values. This automated trending analysis program could then assign symbolic trending values, such as "noisy", which would be used as input to the diagnostic reasoning process. In some cases, the operator may be asked to assess the stored values manually or to review the trend determination made by the system. We envision a strip-chart-like display which would be made available for the user so as to allow him to help determine the trend of the data.

We are currently planning to employ a neural network approach to determine the telemetry trends. While it may have been possible to implement trending analysis using ART-IM rules, it was thought that the addition of trend determination rules would overly complicate the BCAUS knowledge base and make it difficult to locate and maintain diagnostic knowledge. In addition, neural networks offer advantages over standard

statistical routines in that they can be readily trained to output not only trends, but symbolic determinations of instrument state such as "noisy gyro 1 A channel", in response to time series telemetry inputs. (See [6] for a similar application.) The training data for the neural network trending process will be based on both expert input and empirically derived results. We hope to have simulated data available for use in the latter.

The neural network simulation package we chose is Neural Works Professional II from Neural Ware, Inc. It will be tied into the inferencing part of the system via C function calls from the ART-IM rules. The C functions will either accept symbolic trend outputs from the neural network trending application for all telemetry items, or will run the application for individual or groups of telemetry items.

We intend to gather all needed telemetry items (currently approximately 200 items) for the 10 to 15 minute time period leading up to and through mode transition. The time period is adjustable; determining the appropriate period is expected to be an iterative process. The data archival process will be a pre-processing step offline from the diagnosis. The archived data will be output to a file before it is input to the telemetry trend processor. It is expected that all telemetry items of interest will be archived from the tape recorder playback data. The actual number of samples to process for trending purposes depends on how often the data is expected to change, and whether the telemetry is continuous analog or discrete.

It is possible that multiple neural networks will be used to perform the trending analysis. To improve the performance of the neural network, a network could be trained for each different telemetry item or type of item. It is also possible for the neural network(s) to learn from mistakes made in the assignment of symbolic trends by using the analyst's corrections to re-train or modify the network.

4.3 User Interface

Plans for the user interface include the incorporation of graphic depictions of the relevant spacecraft subsystems in the form of functional block diagrams and causal graphs with potential problem areas highlighted. These will be presented in hierarchical levels allowing movement between top-level subsystem overviews and lower, more detailed, component views.

The user interface has become complicated by the recent shift in project goals away from providing an immediately operational system and toward providing a system which is ready to populate with GRO-specific data, which would then be operational. This latter aim is more complex, because it means that ease of maintenance is even more important than before. As a result, we are considering how to devise easy ways for the GRO FOT to amend the knowledge base, re-train the neural network, and update the graphics interface as needed. Ideally, a tool-kit would be provided to aid the FOT in making the necessary additions and changes; realistically, the funding to do this additional work is not available. Thus, it is of primary importance to consider how modularization of functionality may be achieved throughout system development, and to provide for effective combinations of manual and automated procedures. The separation of the trend analysis and diagnostic functions was motivated by the objective of providing a user-maintainable system.

5 Discussion

The experience of working with tools as different in approach as KES HT and ART-IM for the same application has forced us to realize that isolation of program control from domain knowledge is a central issue in our application. Thus we have had to consider some basic knowledge representation and reasoning issues having to do with the separation of domain and reasoning knowledge, the form with which to best represent these kinds of information, and the advisability of integrating specialized components to accomplish an "expert system" task.

Knowledge-based systems are defined in part by their separation of domain knowledge from program control. Expert systems are a subset of knowledge-based systems that exhibit extensive expertise in a particular domain. The separation of domain knowledge from control of the reasoning process allows the

domain knowledge to be more explicit and accessible. If the control and domain knowledge are intermixed, it becomes less clear as to what should be changed to correct or improve the system. It is also less clear as to what the side-effects (if any) of such changes might be. The result is a less flexible and maintainable system.

In simple production rule systems, the domain knowledge is encoded in rules and manipulation of the domain knowledge is handled by the inference engine. This separation is what allows expert system shells to be marketable; developers supposedly have only to code the rules specific to their application, and the shell supplies the general rule reasoning capability. In actual use, however, the separation of domain knowledge and manipulation of that knowledge is not clean. Expert system shell manufacturers find it necessary to provide ways to encode *meta-knowledge*, that is, information, usually procedural, about what to do with the domain knowledge. The shells try to provide the iteration and sequencing control that pure rule-based systems eliminate. However, sequencing rules with such control mechanisms as rule priorities, state variables, and an agenda is more complex and has been likened to programming via side effects [7].

Using KES HT, we built a prototype in which no explicit rules were coded. This is because KES HT supplies a built-in diagnostic inference engine and a frame-based method of representing causes and symptoms. The frames effectively captured the causal rules by implicitly representing the relationship between a failure and possible manifestations. As a result there were no rules to maintain and the knowledge representation was very accessible and independent. Using a shell specializing in diagnosis provided near-complete isolation of program control, which was confined to the inference engine. The domain knowledge alone comprised the knowledge base.

Davis and King [7] discuss applications where the knowledge to be encoded has a strongly sequential character. They state that the inference process can be viewed as a passage through a sequence of states. To control the sequence of rule firings, the system states are tracked with identifiers which are used in the "if" portion of the rule. To fire the appropriate rule, the corresponding state is checked. The resulting rules may look independent as they appear to be individual inference steps, but they are in fact locked together in a tight structure, a sequence of state transitions that defines what to do next. The removal or addition of a rule could destroy the sequence. The point is that some information is inherently sequential. Sometimes we want to know that after we have done W, do X, then do Y, then do Z. The knowledge in such a domain is knowledge of the correct sequence of actions. In such cases the inference engine and knowledge base become nearly indistinguishable.

Part of the dilemma harkens back to early debates in the field about whether expert systems were supposed to encode the "what" or the "how" knowledge. (See [8] for a review of that debate.) If we wish to reason in a particular manner (e.g., minimal set covering), then we must either use an inference engine designed for that purpose or must construct rules (or whatever) to produce the same effect. If we undertake to do the latter, then the problem becomes maintenance of "what" and "how" knowledge separation. One way to simplify this problem is to localize control by breaking the knowledge base into modules; another is to make use of a set of cooperating tools. In the BCAUS project, our approach to knowledge base organization has been to use the ART-IM schemas to encode the "what", or declarative knowledge, and the rules to encode the "how", or procedural knowledge. We have also moved the telemetry trending analysis function to a specialized tool, the neural network. We think that these decisions will help to keep the knowledge base smaller and will facilitate maintenance by restricting the changes the FOT will need to make to the relatively readable schemas. The addition of the neural network actually decreases the control problems of diagnosis, and the strengths of the neural network technology appear very well-suited to the application domain.

6 Conclusions

A number of approaches were used that helped us to achieve some measure of separation between knowledge and control. One approach is to use special-purpose inference engines such as the KES HT minimal set covering diagnostic inference engine. Specialized systems and shells incorporate additional

control because they know what kind of control is needed for a particular application. Another approach is to remove the procedural aspects of the problem from the inference engine and employ appropriate algorithms or techniques, using a conventional procedural language or multiple specialized components. Yet another approach is to employ an expert system tool or language that supports procedural constructs within the rules. This is preferable to sequencing rules through the use of state variables.

Our experience on this project has led us to conclude that specialized shells, such as KES HT, may be the best choice for some applications that closely match the shell's design. Specialized shells allow a definite separation between domain knowledge and control, with the latter residing solely in the inference engine. If, however, circumstances force the use of a more generic tool, weaker in terms of built-in application-specific control, the implementor can try to explicitly separate the control aspects from the domain knowledge by making use of the constructs provided by the shell. This is what we have attempted to do in ART-IM by restricting control to the rule set, and encoding domain knowledge in the schemas. Alternatively, or in conjunction with this approach, the system can be broken into a number of cooperating, specialized tools. This permits both further isolation of control, such as use of a C program driver and embedded calls to C programs, and limits the scope of the shell's responsibilities. We are trying to achieve this by using a neural network front-end to perform a specialized trending analysis function, a C program to provided overall control and user interface, and the expert system shell to perform and explain the diagnostic reasoning process.

Approaches such as these that promote the separation of knowledge and control are described in current research and project reports. Rarely acknowledged, however, is that the truism, that knowledge and control are segregated in expert systems, is difficult to obtain. As a result, it is partly responsible for disappointment and misunderstandings about expert systems development and maintenance. The separation of knowledge and control is a goal to aim for rather than a sure foundation upon which to build. Recognition of this fact should help to foster realistic expectations among both developers and users about expert systems and maintain their credibility as a problem solving approach.

ACKNOWLEDGMENTS

The authors wish to thank the project Assistant Technical Representative, Dan Mandl, Code 511.2, Goddard Space Flight Center for his encouragement of this effort. We would also like to thank our supervisor, Doug Carlton, for his assistance and helpful suggestions.

REFERENCES

[1] Computer Sciences Corporation (1988). *Gamma Ray Observatory Backup Control Mode Analysis and Utility System (BCAUS) Prototype Requirements, Design, and Implementation*, CSC/TM/88-6069, prepared for Goddard Space Flight Center.

[2] Jerkovsky, W., L. Keranen, F. Koehler, F. Tung, B. Ward (1986). "GRO Attitude Control and Determination", 86-032, Annual Rocky Mountain Guidance and Control Conference, Keystone, Colorado.

[3] Tai, F., N. Kaskak, K. McLaughlin, B. Ward (1987). "An Innovative Design for Autonomous Backup Attitude Control of the Gamma Ray Observatory", 87-2601, AIAA Guidance Navigation and Control Conference, Monterey, California.

[4] Reggia, J., D. Nau, P. Wang (1983). "Diagnostic Expert Systems Based on a Set Covering Model", International Journal of Man-Machine Studies, 19.

[5] Reggia, J., Y. Peng (1986). "Modeling Diagnostic Reasoning: A Summary of Parsimonious Covering Theory", Proceedings of Computer Applications in Medical Care 1986, Washington, D. C.

[6] Bell, B. and J. L. Eilbert (1988). "Controlling Basins of Attraction in a Neural Network-Based Telemetry Monitor", Fourth Conference on Artificial Intelligence for Space Applications, NASA Conference Publication 3013.

[7] Davis, R., King, J. (1975). "An Overview of Production Systems", Stanford Artificial Intelligence Laboratory, Memo AIM-271.

[8] Winograd, T. (1975). "Frame Representations and the Declarative/Procedural Controversy", 185-210, Representation and Understanding: Studies in Cognitive Science, ed. D. G. Bobrow and A. M. Collins, New York: Academic Press.